Semantics as a Functor in LFG+Glue

Avery D Andrews, ANU, Dec 12 2008
draft; please do not cite without permission
(minor revisions from earlier Dec draft)

In this brief note, I will present a way to present semantic interpretation in LFG glue semantics (Dalrymple (2001), Asudeh (2004), and many other works) as a category-theoretic functor, somewhat in the spirit of Dougherty (1993) for Categorial Grammar, but with substantial differences due to the rather different environment provided by LFG+glue. The basic idea is that glue proofs naturally determine a small Cartesian Closed Category (CCC), while many proposals for model-theoretic semantics present model-theoretic interpretations as inhabitants of a (usually not small) CCC. Then, if semantic interpretation is required to be a CC functor, it doesn't require much additional work to set it up.

In particular, all we have to do is:

(1) a. Specify the 'real semantic type' (object in the real semantics CCC) corresponding to each 'glue semantic type' (basic semantic type symbol used in the glue system). These are usually e and t in pedagical presentations, although here I will tend to use E and P).

b. Specify CCC arrows in the real semantics category for such closed category words as are to be defined (e.g. pronouns and other 'logical words').

This is quite modular, since the only thing that the syntax and the semantics have to know about each other is the function connecting the glue semantic types to the real ones.

A question one might ask is, why bother with LFG+glue, when category-theory based analysis is already essentially present in, for example, categorial grammar and its variants? The answer to this is that LFG has a fairly stable body of analyses of many typologically diverse languages, including a significant number of large 'industrial strength' grammars, as well as descriptive treatments of a variety of 'exotic' phenomena, which don't yet seem to have treatments in the categorial-grammar-based frameworks. It therefore seems worthwhile to explore ways of easily attaching CG-compatible semantics to LFG analyses, if only to get a better sense of what the true differences between frameworks amount to.

The category theory is presented in a fairly barebones manner, see basic textbooks plus Lambek and Scott (1986) (LS) and Troelstra and Schwichtenberg (2000) (especially ch. 8) for more.

1 Grammatical Background

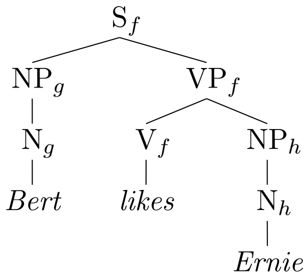
I will first briefly present LFG syntactic structures. These consist of a phrase-structure tree, called a 'c-structure', which states the parts of speech and phrase-types of an utterance, in their overt linear order,[1] and the 'f-structure', which presents various further information

[1] With a very few possible exceptions, such as 'second position clitics', where members of a finite list of words can appear once constituent after where they would be expected to appear on the basis of how the syntax as a whole works.

about the grammatical structure, such as grammatical relations and inflectional features (similarly to traditional grammar, with some differences). These two are related by a function ϕ, which assigns to each (non-terminal) node in the c-structure tree a correspondent in the f-structure.

An example is the following, where ϕ is represented by the labels subscripted to the c-structure nodes, and placed in front of f-structures, separated by colons:

(2) a.



b.

$$f{:}\begin{bmatrix} \text{SUBJ} & g{:}\begin{bmatrix}\text{PRED} & \text{`Bert'}\end{bmatrix} \\ \text{TENSE} & \text{PRES} \\ \text{PRED} & \text{`Like(SUBJ, OBJ)'} \\ \text{OBJ} & h{:}\begin{bmatrix}\text{PRED} & \text{`Ernie'}\end{bmatrix} \end{bmatrix}$$

For more on the architecture, see for example Kaplan (1995).

Many aspects of grammar can be nicely accomodated with LFG, but formal semantics has tended to be problematic, in part because the f-structures don't have to be trees (reentrancies are definitely allowed, and maybe even cycles), which subverts the operations of the bottom-up tree-climbing methods often used for semantic interpretation, and also because they are often 'too flat'. For example the f-structure of a sentence such as *Everybody probably loves somebody* would have overall the same form as (2), with an additional 'ADJUNCT' attribute added to house the adverb, and no account of all of the numerous possible scoping relationships between the quantifier and the adverb. Although many different proposals for semantic interpretation of LFG have been made over the decades, they have tended to have a very limited following, together with difficulties with some fairly standard data; for example the proposal of Halvorsen and Kaplan (1988) does not seem to have any way of managing the two scopes of *everybody loves somebody.*

2 Glue

'Glue semantics' is an application of the implication-tensor fragment of intuitionistic linear logic, often called 'MILL' (for 'multiplicative ILL', however the usage of this term is inconsistent as to whether units and exponentials are included), which manages to solve the problem of interpreting f-structures.[2] The first basic idea is that the words and possibly other meaningful elements (formatives within words, and perhaps certain phrase-structure configurations) of the sentence provide a collection of 'meaning-constructors', which are the

[2]And can be applied to various other kinds of syntactic formats as well, as reviewed in Asudeh (2004).

semantic contributions from which the meaning of the whole utterance is to be assembled. For a first approximation, the meaning-constructors can be thought of as pairings of an actual meaning with a semantic type, the latter made out of basic types and an implicational type-constructor, and perhaps a product as well (there is some controversy about whether the product is needed in the meaning-constructors). Since the logic is linear, the product, if present, will behave like a tensor rather than a cartesian product, in the glue derivation.

Therefore, if we have semantic types E for 'entity' and P for 'proposition', the meaning-constructors for *Bert likes Ernie* might be (assuming the usual convention of omitting rightmost parentheses with implications):

(3)
$$\begin{array}{ccc} E & E & E \multimap E \multimap P \\ \mathit{Bert} & \mathit{Ernie} & \mathit{Like} \end{array}$$

Note that the verb is treated as 'curried'; linguists find various motivations[3] for supposing that that the arguments of verbs are organized in a hierarchy with 'least active' corresponding to 'applied first' in a curried scheme. So the first E of *Like* represents the Likee, the second, the Liker.

The motivations for currying are widely but not universally accepted; dissidents can and sometimes do use products/tensors rather than curries (assuming the usual implication that product-like operators bind more tightly than implications):

(4)
$$\begin{array}{ccc} E & E & E \otimes E \multimap P \\ \mathit{Bert} & \mathit{Ernie} & \mathit{Like} \end{array}$$

With tensoring of the arguments, the more active one would conventionally appear first.[4] Since the logic of combination is (commutative) linear, the constructors form a multiset, so that relative order is not significant, but multiplicity of occurrence is.

There are two possible ways of combining the constructors of (4), which correspond to different linear logic deductions of a type P conclusion from them. These can be conveniently represented by 'proof-nets', where the type P final conclusion is separated by a turnstyle from the assumptions, and the curved lines are 'axiom links', which represent applications of the identity axiom in the Gentzen sequent calculus:

(5) a.
$$\begin{array}{cccccc} E & E & E \multimap E \multimap P & \vdash & P \\ \mathit{Bert} & \mathit{Ernie} & \mathit{Like} & & \end{array}$$

b.
$$\begin{array}{cccccc} E & E & E \multimap E \multimap P & \vdash & P \\ \mathit{Bert} & \mathit{Ernie} & \mathit{Like} & & \end{array}$$

Axiom-links must pair up the literals of the formulas with no overlaps or omissions; this is sometimes called 'perfect matching'.

[3] Such as the 'Thematic Role Hierarchy' of Jackendoff (1973), and the idiom-structure argument of Marantz (1984).

[4] For roughly the same kinds of reasons that motivate the layering of arguments in the curried scheme, following a correspondence of linear 'earlier' with layered 'more outer'.

Added axiom-links are probably as close as one can get to a minimal-effort depiction of how meaning to be assembled (I conjecture that they might might be useful representations for the grammatical structures required for defining notions such as 'intensional isomorphism'), but, unfortunately, it is not the case that just any perfect matching represents a valid proof; various constraints must be satisfed, by processes that involve a bit of work.

One is a rather obvious constraint that the semantic types of paired literals must be the same, another depends on a property called 'polarity', with values +/−, introduced by Jaśkowski (1963), and defined as follows:

(6) a. the polarity of a formula on the left of the turnstyle is negative

b. the polarity if a formula on the right of the turnstyle is positive

c. The polarity of the components of a tensor is the same as that of the whole tensor

d. The polarity of the antecedent of an implication is opposite to that of the whole implication

e. The polarity of the consequent of an implication is that same as that of the whole implication

So the next constraint, which is easy to check on the structures (4), is that paired literals must have opposite polarity.

The final condition, called the 'Correctness Criterion', is more complex, and has a rather remarkable number of different-looking but mathematically equivalent formulations.[5] The one that I find most congenial for linguistics comes from de Groote (1999), and is based on a concept called the 'dynamic graph'.[6] The dynamic graph is a directed graph connecting the nodes of the subformula trees of the formulas in the sequent. It can be viewed as a partial reorganization of the syntax-trees of the formulas combined with the axiom-links. The link-building depends on polarity, as follows:

(7) a. An axiom link is a dynamic graph-link directed from negative to positive

b. In a negative implication, there is one dynamic graph link from the entire implicition to its consequent, and another from the antecedent to the consequent

c. In a positive implication, there is a dynamic graph link from the consequent to the entire implication.

The effects of the dynamic graphs rules can be conveniently represented like this:

(8)

$$\begin{tikzcd}[column sep=small] A^- & & B^- \\ & (A \otimes B)^- \arrow[ul] \arrow[ur] & \end{tikzcd} \qquad \begin{tikzcd}[column sep=small] A^+ \arrow[dr] & & B^+ \arrow[dl] \\ & (A \otimes B)^+ & \end{tikzcd}$$

$$\begin{tikzcd}[column sep=small] A^+ \arrow[r] & B^- \\ (A \multimap B)^- \arrow[ur] & \end{tikzcd} \qquad \begin{tikzcd}[column sep=small] A^- & B^+ \\ (A \multimap B)^+ \arrow[ur] & \end{tikzcd}$$

[5]Moot (2002:ch. 4) reviews several of them in a clear and accessible manner.

[6]Which is essentially the same thing as the 'essential net' of Lamarche (1994).

Note that a negative antecedent isn't linked to any other node of its (positive) implication.

Andrews (2008a) develops the point that constructing the dynamic graph links is essentially the same thing as building a conventional linear lambda-term for the assembly, with the original proof-net links from positive implications to their (negative) antecedent (represented only implicitly in (8), by the arrangement of the nodes) representing variable-binding.

The Correctness Criterion can then be stated in terms of the dynamic graph:

(9) The Dynamic graph must:[7]

a. be acyclic, and
b. from any negative polarity (sub)formula-occurrence on the left of the turnstyle, provide a path to the (root node of) the formula-ocurrence to the right of the turnstyle
c. such that any path starting at a negative antecedent (of a positive implication) passess through the consequent of that implication.

If there aren't any tensors, the requirement is a bit simpler: the dynamic graph must form a tree. For linguistic purposes, the representation is quite neat until one comes to bound pronouns, for which tensors provide the most worked out analysis (Asudeh 2004); then things get messier, but not, I believe, untenable, as discussed in Andrews (2008a).

However we haven't accounted for how the grammatical information constrains the semantic assembly, which is where the f-structures come in. While Categorial Grammar and its relatives use word order directly to limit the possibilities, in LFG+glue, the grammatical analysis produces what can be seen as an enrichment of the types with information about grammatical location in the sentence's f-structure. In our current example, *Bert* and *Ernie* would be associated with the f-(sub)structures g and h ('subject' and 'object' substructures), while the verb, verb-phrase and entire clause are associated with the entire f-structure f.

We can use these associations to specify an additional tier of information that the proof-net linking must be consistent with, on the basis that in order to be connected by axiom links, literals must have the same f-structure location as well as semantic type. So the meaning-constructors become:

(10)
$$\begin{array}{ccccc} g & h & h \multimap g \multimap f & & f \\ E & E & E \multimap E \multimap P & \vdash & P \\ \textit{Bert} & \textit{Ernie} & \textit{Like} & & \end{array}$$

For which only the (b) linking above is now possible, given the f-structure matching requirement on axiom-links. In this particular example, the f-structure information determines a unique linking, but there are also cases where the semantic types are needed to correctly constrain linking,[8] and where multiple assemblies are possible, as we'll see in a moment.

The meaning-constructors thus need to specify f-structure information as well as semantic information. There is more than one method whereby this can be achieved; see Dalrymple

[7]These are equivalent to the algebraic criterion that deGroote proposes, although the equivalence is not worked out explicitly.

[8]Dalrymple p.c., discussed in Kokkonidis (2008).

(2001), Asudeh (2004) or Andrews (2008a) for the standard view, and Andrews (2008b) for an alternative. We will here simply assume that something is giving us meaning-constructors with the content of (10), and proceed from there. Another point is that while glue standardly uses universal quantification to account for certain kinds of scope variation, here we dispense with this, and instead use LFG instantation, as discussed in Andrews (2008a).

We'll close this section by showing how the account deals with the scope ambiguity of sentences such as (11), which LFG f-structures alone don't adequately represent:

(11) everybody didn't leave

In the f-structure, we'd have a single layer with the *everybody* content sitting in subject position, and the negativity as the value of a 'polarity' feature (emphatic positive polarity would be expressed as *everybody did so leave!*):

(12)
$$f{:}\begin{bmatrix} \text{SUBJ} & g{:}\begin{bmatrix}\text{PRED `Everybody'}\end{bmatrix} \\ \text{TENSE} & \text{PAST} \\ \text{POLARITY} & \text{NEG} \\ \text{PRED} & \text{`Leave(SUBJ)'} \end{bmatrix}$$

This provides no structural basis for the ambiguity, and neither does the phrase-structure.

But with the meaning-constructors, things are a lot better. Ignoring tense (which everybody does in basic discussions, since tense is rather hard), the constructors would be (treating *Everybody* as being a property of properties, that is, a property that is true of all properties that are true of all people[9]):

(13)
$$\begin{array}{ccccc} g \multimap f & (g \multimap f) \multimap f & f \multimap f & \vdash & f \\ E \multimap P & (E \multimap P) \multimap P & P \multimap P & & P \\ \mathit{Leave} & \mathit{Everybody} & \mathit{Not} & & \end{array}$$

This has two possible linkages, subject to the constraints we've imposed, one where *Everybody* take scope over the negative, the other under it. The same principle will work for two-quantifier ambiguities such as that of *everybody loves somebody.*

But (12), with its negative marker, also shows why we need to use linear rather than relevant or full intuitionistic logic. For with intuitionistic logic, we could interpret a negative sentence as a positive one by ignoring the negative, while with relevant logic, we could do the same thing by interpreting it twice (building proof-terms as logical forms with the Curry-Howard correspondence).

So how do we get to the meanings? Proof-nets are one of a number of techniques available for representing normal proofs, and the way these are standardly used to produce meanings is that the Curry-Howard Isomorphism is used to determine a linear lambda-term on the basis of the proof. For example, the terms for the two readings of (13) would be:[10]

[9] This is the widely accepted 'generalized quantifier' analysis of Barwise and Cooper (1981)

[10] See de Groote and Retoré (1996) and Perrier (1999) for some concrete techniques for building lambda-terms from proof-nets.

(14) a. $\mathit{Everybody}(\lambda X.\mathit{Not}(\mathit{Leave}(X)))$

b. $\mathit{Not}(\mathit{Everybody}(\lambda X.\mathit{Leave}(X)))$

These lambda-terms are then assumed to undergo β-reductions with whatever the internals of the meaning-sides are supposed to be, and this is where the theory becomes rather unconstrained, since the temptation to get clever with contents of the meaning-sides is extremely difficult to resist.

The proposal here is to proceed in a superficially somewhat different manner. First, the proof/net glue-proof, which is, among other things, an arrow in a Symmetric Monoidal Closed Category (SMCC), is taken as also being an arrow in the (small) Cartesian Closed Category (CCC) generated by the literals, with the SMCC tensor and implication as product and implication, respectively (so that what we're really doing is adding pairing and projections). Semantic Intepretation is then construed as a functor from the small CCC into whatever CCC has been chosen to be the 'Real Semantics' category, which, for standard model-theoretic semantics, would be **Sets**, but could be anything else that has been set up as a CCC, such as for example Pollard's (2008) CCC of hyperintensions. This functor says what the Real Semantics counterpart of the small CCC's CC apparatus is, and the image under the functor of the small CCC;s proof-net arrow is the interpretation of the sentence.

3 Making a small CCC

It is a standard result that by equating certain combinations of proof-steps, proofs in various logics, including the one used in glue, can be construed as arrows in a category. A mildly annoying fact is that for this to work, the premises of the sequent need to be packed into a single category-theoretic object with a definite ordering imposed, as well as association, which will be arbitrary if the logic is commutative (and associative). We then use a 'combinatorial' version of the logic, in which all sequents are of the form $A \vdash B$. This doesn't seem to create issues for applications to linguistics, although does provide some subtle talking-points.[11] If the logic is MILL, the category is a 'symmetric monoidal closed category'[12] (SMCC), and if the application is glue semantics, there is no need to worry about the 'units', which avoids quite a bit of fuss and worry.

However, the definition of an SMCC is somewhat complex and tedious, and as long as the Real Semantics is going to be a CCC, we can save some trouble by construing the proof-net as a small CCC immediately. This works because of a theorem originally proved by Babaev and Solov'ev (1982), with a recent pedagogical presentation in Troelstra and Schwichtenberg (2000:ch 8), which says that a 'balanced' intuitionistic implication-conjunction sequent, that is, one in which each proposition-letter occurs only twice, has at most one normal proof. This means that in the corresponding CCC, there will be at most one arrow connecting a conjunction of the premises to the concusion, and, since linear tensor-implication logic is a subset of intuitionistic conjunction-implication logic, if a balanced linear sequent is linearly valid, it won't acquire any additional (essentially different) proofs when we move to the more

[11] As discussed for example at `http://golem.ph.utexas.edu/category/2008/02/logicians_needed_now.html`.

[12] See Troelstra (1992), Melliès (2008); the math level of the latter rises inexorably as it proceeds, but the earlier sections are quite accessible and full of useful information.

permissive full intuitionistic system.

So, from the proof-net, construed as a balanced sequent, we construct a small CCC along the lines set out by LS:47-87 and elsewhere. Later, we'll briefly give a reason for not dwelling on the SMCC.

For the basic objects of this CCC, we essentially use the f-structure labels of the glue sequent, supplemented with some additional marker such as a superscript to distinguish members of different axiom-linked pairs, but to avoid distracting clash of notation, we will recode them as distinct upper case letters, and also replace the standard SMCC tensor and implication symbols with CCC product and implicatin symbols. The result for the wide-scope reading for the quantifier in (15) is:

(15)
$$\begin{array}{ccccc} A\Rightarrow B & (A\Rightarrow C)\Rightarrow D & B\Rightarrow C & \vdash & D \\ g\multimap f^1 & (g\multimap f^2)\multimap f^3 & f^1\multimap f^2 & \vdash & f^3 \\ E\multimap P & (E\multimap P)\multimap P & P\multimap P & & P \\ \textit{Leave} & \textit{Everybody} & \textit{Not} & & \end{array}$$

In addition to the objects generated from the basic ones by the $\times$ and $\Rightarrow$ operations (here I reverse the left-right orientation used by LS), the CCC will also contain a terminal object **1**.

There are also the standard CCC arrows:

(16) a. composition: for each $f{:}A \to B, g{:}B \to C$, $gf{:}A \to C$.

b. identities: for each A, $\mathbf{Id}_A{:}A \to A$.

c. terminals: for each A, $\bigcirc_A{:}A \to \mathbf{1}$.

d. pairing: for each $f{:}A \to B, g{:}A \to C$, $<f,g>{:}A \to B{\times}C$

e. projections: for each A, B, $\pi^1_{A,B}{:}A{\times}B \to A$ and $\pi^2_{A,B}{:}A{\times}B \to B$

f. evaluation: for each A, B, $\mathbf{ev}_{A,B}{:}(A{\Rightarrow}B){\times}A \to B$

g. curry: for each $h{:}C{\times}B \to A$, $\mathbf{cur}(h){:}C \to B{\Rightarrow}A$.

Next, some equations:

(17) a. for $f{:}A \to B, g{:}B \to C, h{:}C \to D$, $(hg)f = g(fh)$

b. for $f{:}A \to B$, $f\mathbf{Id}_A = \mathbf{Id}_B f$

c. If $f{:}A \to \mathbf{1}$, then $f = \bigcirc_A$ (arrows to the terminal object are unique).

d. for $f{:}C \to A, g{:}C \to B$, $\pi^1_{A,B}<f,g> = f$

e. for $f{:}C \to A, g{:}C \to B$, $\pi^2_{A,B}<f,g> = g$

f. for $h{:}C \to A \times B$, $<\pi^1_{A,B}h, \pi^2_{A,B}h> = h$

g. for all $h{:}C \times B \to A$, $\mathbf{ev}_{A,B}<\mathbf{cur}(h)\pi^1_{C,B}, \pi^2_{C,B}> = h$

h. for all $k{:}C \to B{\Rightarrow}A$, $\mathbf{cur}(<k\pi^1_{C,B}, \pi^2_{C,B}>) = k$

The arrows of the free CCC over the proposition letters are the equivalence classes of the intersection of all equivalence relations over the arrow-expressions that satisfy the equations.

Then, the fact that there is a linear proof of the sequent shows that there is an intuitionistic one as well, and the B&S theorem tells us that this proof is unique, while the construction of proofs as arrows in a free CCC tells us that the corresponding arrow in the free CCC is also unique. But what we want our semantics to yield is an arrow from **1** to the propositional type in the semantics category (that is, an element of the propositional type), so if semantic interpretation is to be a functor, we need such an arrow here.

This can be provided by the names that appear on the meanings-sides of constructors, such as '*Everybody*', which can be construed as arrows from **1** to the propositional formula derived from their meaning-side, e.g., in (15), *Everybody* is an arrow from **1** to $(A{\Rightarrow}C){\Rightarrow}D$. LS describe how to add such 'indeterminates' to a free CCC. Given these, we can then construct an arrow from **1** to some product of (all of) the premises, and compose this with the arrow determined by the proof to get our desired arrow from **1** to P.

But we certainly need this to be insensitive to the exact product of the premises that is chosen to represent the proof-net, and, ideally, a unique arrow from **1** to the proposition letter of the conclusion of the sequent in the small CCC, extended by the indeterminates. This follows as a consquence of Corollary 6.2 of the Functional Completeness Theorm of LS:59-61.

The exact result we need is:

(18) Let $\mathscr{C}$ be a CCC, $A = \prod A_i$ an object of $\mathscr{C}$, $p{:}A \to C$ the unique arrow in $\mathscr{C}$ from A to C, and $a_i{:}\mathbf{1} \to A_i$ 'indeterminates' in the sense of LS:57, adjoined to $\mathscr{C}$ to produce a new category $\mathscr{C}'$. Then $\mathscr{C}'$ contains an arrow $a{:}\mathbf{1} \to A$, and pa is the unique arrow in $\mathscr{C}'$ from **1** to C.

Proof: The indeterminates trivally combine by means of pairing to produce $a{:}\mathbf{1} \to A$, and $\mathscr{C}' \approx \mathscr{C}[a]$. Now let $c{:}\mathbf{1} \to C$ be an arrow of $\mathscr{C}[a]$. By Corollary 6.2, there is a unique arrow $q{:}A \to C$ in $\mathscr{C}$ such that $c = qa$, but since p is unique from A to C in $\mathscr{C}$, $c = pa$. QED.

It might be useful to explicitly calculate this resulting arrow for (15). This can be done more smoothly with the aid of some notations defined in LS:

(19) a. name of: for $f{:}A \to B$, $\ulcorner f \urcorner{:}\mathbf{1} \to A{\Rightarrow}B = \mathbf{cur}(f\pi^2_{\mathbf{1},A})$

b. 'func of': for $n{:}\mathbf{1} \to A{\Rightarrow}B$, $n^{\int}{:}A \to B = \mathbf{ev}_{A,B}<n\bigcirc_A, \mathbf{Id}_A>$

(These are easily shown to be inverse to each other). The composition of (what we think of as the intuitive meanings of) *Not* with *Leave* will now be $\mathit{Not}^{\int}\mathit{Leave}^{\int}{:}E \to P$, which composes with $\mathit{Everyboy}^{\int}$ to yield:

(20) $\mathit{Everybody}^{\int}\ulcorner \mathit{Not}^{\int}\mathit{Leave}^{\int} \urcorner{:}\mathbf{1} \to P$

Systematic techniques for constructing such arrows can be obtained by combining proof-net 'semantic reading' methods with the Functional Completeness theorem, which assures that the 'variables' introduced by the reading techniques can be removed. As described in LS, the arrows can also be represented as ordinary-looking lambda-terms, with the Functional Uniqueness theorem saying that, appearances to the contrary, variables aren't really involved.

The desired uniqueness of the arrow from **1** is the main reason for using the freely generated CCC rather than SMCC on the proposition letters: the literature on SMCCs doesn't contain any straightforward techniques for producing such an arrow.

4 Semantic Interpretation as a Functor

A functor $F{:}\mathscr{C} \to \mathscr{D}$ from category $\mathscr{C}$ to $\mathscr{D}$ is something that assigns to each object of $\mathscr{C}$ an object of $\mathscr{D}$, and to each arrow of $\mathscr{C}$ an arrow of $\mathscr{D}$ such that the following conditions are satsified:[13]

(21) a. If $f{:}A \to B$ is an arrow of $\mathscr{C}$, then $F(f){:}F(A) \to F(B)$.

b. If A is an object of $\mathscr{C}$, then $F(\mathbf{Id}_A) = \mathbf{Id}_{F(A)}$

c. If $f{:}A \to B, g{:}B \to C$ are arrows of $\mathscr{C}$, then $F(g \circ f) = F(g) \circ F(f)$.

A functor is furthermore Cartesian Closed if goes from one CCC to another, perserving all of the CC apparatus.[14]

So to set up a standard baby Montague Grammar-style extensional grammar, we first need to present the category of sets and their functions (**Sets**), as a CCC, which is done by specifying what the various objects and arrow are, as follows:

(22) a. terminal object: any one-element set, standardly written as $\{*\}$.

b. product object: standard Cartesian Product. If we need to distingish the **Sets**-specific product from the product in an arbitrary CCC, we can write the former as $\times_{\mathbf{Sets}}$.

c. implication object: function space. That is, if A, B are sets, then $A{\Rightarrow}B = B^A$.

d. terminal arrows: For each set A, $\bigcirc_A$ is the function from A to $\{*\}$ that takes $a \in A$ to $*$.

e. projection arrows: For each pair of sets A, B, $\pi^1_{A,B}$ and $\pi^2_{A,B}$ are the functions from $A \times B$ to A and B that take $<a, b>$ to a and b, respectively.

f. pairing arrow: For each pair of functions $f{:}C \to A, g{:}C \to B$, $<f, g>$ is the function from C to $A \times B$ that takes c to $<f(c), g(c)>$.

g. evaluation arrow: for sets A, B, $\mathbf{ev}_{A,B}{:}(A{\Rightarrow}B) \times A \to B$ is the function that takes $<f, a>$ to $f(a)$.

[13] In textbooks, the object and arrow portions of a functor are usually distinguished by the subscripts 0 and 1 when functors are introduced, but this seems pointless to me, and probably to the authors as well, since the convention is usually abandoned in practice almost immediately.

[14] For us, 'strictly.' But mathematicians seem to be more interested in 'lax' functors that only preserve structure up to natural isomorphisms. But I can't see how the lax concepts could be relevant to this linguistic application.

h. curry operation on arrows: for each function $h{:}C \times A \to B$, $\mathbf{cur}(h)$ is the function from C to B^A such that for each $c \in C$, $\mathbf{cur}(h)(c)(a) = h(c, a)$.

This setup satisfies the equations (17). It may seems like quite a lot of stipulation for people who haven't seen anything like it before, but it is all standard textbook stuff, not devised for any specifically linguistic purpose.[15] And the final moves to produce the semantics are then very simple:

(23) a. specify the sets corresponding to the E, P glue-semantic types

b. specify the images under the semantics functor of any meaning-sides whose interpretation we want to fix (others are left freely variable, in the usual way of nonlogical constants in model theory).

For a standard extensional semantics, (a) will be accomplished by setting $F(E) = U$, where U is some chosen 'universe of discourse', and choosing some set such as $B = \{\mathbf{T}, \mathbf{F}\}$ to serve as the truth-value, and stipulating that $F(P) = B$.

For (b), there is a difference between meaning-sides that can be specified in any CCC, versus those that depend on specific facilities of whatever we have chosen to be the real semantics category. Bound anaphoric pronouns as analysed by Asudeh fall into the former category; we can represent them either as arrow-terms ($<\mathbf{Id}_{F(E)}, \mathbf{Id}_{F(E)}>$) or as closed typed lambda-terms ($\lambda x^{F(E)}.[x, x]$). The types here here specified in terms of the interpretation functor F, so as to be independent of what the semantics category is to be; in a full meaning-constructor, this information can be supplied by convention from the glue-side).

Other meaning-sides that commonly appear, such as for quantifiers, involve logical constants and various other bits of machinery that make more specific demands on the nature of the semantics category; most of what appears in practice seems like it would exist in a topos, although this has not been worked out explicitly.

So, if we want to use Pollard's (2008) hyperintensional semantics, but use LFG+glue instead of Higher Order Grammar in its entirety, we can just say that $F(E) = \text{Ind}$, and $F(P) = \text{Prop}$, since this semantics comes already set up as a CCC (in fact, a topos). In terms of what goes on with a set-theory based model, the treatment really isn't any different from a Montague-grammar-based one, except that a considerable amount of the work is done by machinery from standard math books, rather than linguistics treatises. And there is furthermore the option to shift the semantics to rather different formal settings, perhaps even things such as Wierzbickian NSM, if these can be set up as CCCs in an appropriate way, with relatively little work on the interface (although the type systems do need to be coordinated).

5 Uses(?) for the Small CCC

The small CCC lets us do our math trick of defining semantic intepretation as a functor, but does it have any specifically empirical function? One point is that it is probably as close as one can get to a finite object that represents a 'reading' of a possibly ambiguous

[15] Although Lambek's long-standing interest in linguistics may have had something to do with those aspects of the formulation that are due to him.

sentence with as little as possible in the way of arbitrary representational choices. A possible intepretation of Jacobson's (1999, 2002, 2005) concept of 'direct compositionality' is that there aren't any such finite objects, but this seems somewhat implausible to me, on the basis that people seem to manipulate them, so they are probably in some sense there. A more generous interpretation is that there should be an absolute minimum of arbitrary-seeming stuff between the overt form of a sentence and a specification of its semantic composition (we do, after all, need at least morphological features such as gender, number and case, as developed by Aristotle and the Sophists, not to mention Paninị). The LFG+glue claim would then be that the syntactic stuff would be the f-structure machinery, empirically motivated on the usual sorts of grounds, such as the typology of how morphology connects to syntax.

It would be better if something of a more concrete nature could be offered; here is a tentative and problematic suggestion. Consider the well-known ambiguity of (24), originally pointed out by Sag (1976):

(24) John$_i$ loves his$_i$ mother, and so does Bill

Even when *his* is taken as coreferential to *John*, it is possible to interpret Bill as loving either John's mother, or his own.

It would be desireable to explain this without postulating that the first clause is structurally ambiguous, which we can do by setting things up in a suitable way. The basic idea will be that a VP-anaphoric expression such as *do so* is of glue-type $(\uparrow\mathrm{SUBJ})_E \multimap \uparrow_P$, meaning that it takes input of semantic type E from its f-structural SUBJ-value, and returns output of semantic type P to its own f-structural location. On the semantic side, however, no value for F is specified directly; rather the preceding discourse is searched for an arrow going from a type-E object linked to a subject of some f-structure to a type-P object linked to that f-structure; the F-value of such an arrow is a possible interpretation for the *do so* arrow.

What we want is for a single reading of an earlier sentence to provide two such arrows. Using Asudeh's (2004) meaning-constructor for pronouns, we can represent the small CCC arrows for the first clause like this, where the superscripts represent f-structures, the subscripts semantic types:

(25)
$$\begin{aligned} \textit{John} &: A^g_E \\ \textit{His} &: A^g_E \rightarrow B^g_E \times C^i_E \\ \textit{Mother} &: C^i_E \rightarrow D^h_E \\ \textit{Love} &: D^h_E \rightarrow B^g_E \rightarrow E^f_P \end{aligned}$$

Note that we're treating the pronominal meaning as an unanalyzed block, since it can't be seen as a pairing of identity arrows until after the semantics functor has applied.

From these materials, we can construct two distinct proofs of the required type (superscripts and subscripts omitted to save clutter, except on the final result):

(26) a.

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathit{John} : A \qquad \mathit{His} : A{\to}B\times C}{\mathit{His}(\mathit{John}) : B\times C}}{\pi^2(\mathit{His}(\mathit{John})) : C} \qquad \mathit{Mother} : C{\to}D}{Mother(\pi^2(\mathit{His}(\mathit{John}))) : D} \qquad \mathit{Love} : D{\to}B{\to}E}{\mathit{Love}(Mother(\pi^2(\mathit{His}(\mathit{John})))) : B^g_E{\to}E^f_P}$$

b.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[X : A]^1 \qquad \mathit{His} : A{\to}B\times C}{\mathit{His}(X) : B\times C}}{\pi^2(\mathit{His}(X)) : C} \qquad \mathit{Mother} : C{\to}D}{Mother(\pi^2(\mathit{His}(X))) : D} \qquad \mathit{Love} : D{\to}B{\to}E}{\mathit{Love}(Mother(\pi^2(\mathit{His}(X)))) : B{\to}E} \qquad \begin{matrix}\vdots \\ \pi^1(\mathit{His}(X)) : B\end{matrix}}{\mathit{Love}(Mother(\pi^2(\mathit{His}(X))))(\pi^1(\mathit{His}(X))) : E}}{\lambda X.\mathit{Love}(Mother(\pi^2(\mathit{His}(X))))(\pi^1(\mathit{His}(X))) : A^g_E{\to}E^f_P}\ 1$$

The ellipsed portion of the (b) proof is identical to the upper left branch, note that the final implication introduction step therefore discharges two occurrences of the assumption.

By well-known contrast, (27) is not generally regarded as having any reading where Bill washes John:

(27) John washed himself, and so did Bill

This might be explained if reflexive pronouns had a somewhat different constructor, discussed by Lev (2007:258), taking this instantiated form:

(28) $\lambda PX.P(X)(X) \ : \ (h_E{\to}g_E{\to}f_P){\to}g_E{\to}f_P$

Lev motivate this form of constructor for reciprocals on the basis of scope argument; these don't apply usefully to reflexives, although the general similarity in binding conditions between reflexives and reciprocals makes it plausible that the overall forms of their meaning-constructors might be similar.

And with additional constructors such as:

(29) $\mathit{Wash} \ : \ h_E{\to}g_E{\to}f_p$
$\mathit{John} \ : \ g_E$

one can easily construct an arrow $\lambda X^{F(E)}.\mathit{Wash}(X)(X)$ to serve as content for the generally accepted reading of *do so* in (27), but not one with meaning $\mathit{Wash}(\mathit{John})$.

The problem I alluded to above, however, is that to determine the number of appropriate arrows that are provided by the small CCC is the rather non-trivial problem of counting normal proofs, discussed for example in Tiede (1999) and Wells and Jakobowski (2003). This is doable, but the methods presented don't seem easy to execute by hand.

6 Possibly Even Wrong

I final remark I'd like to make is that the claim that semantic interpretation is a CC functor is possibly even wrong, therefore perhaps getting over the hurdle of empirical vacuousness. A considerable problem lies in the treatment of intersentential anaphora, where there is a rather long history (reviewed in Lev (2007)) of augmenting glue with some sort of DRS-theoretical device. These proposals involve supplementing the strictly CCC aspects of the meaning-side with some kind of additional machinery, emulating DRS theory, in a way that hasn't yet been given any clear mathematical specification. It isn't clear whether they will fit into the CCC idea, but at least they clearly require more than the basic CCC ideas to function. It also remains to work out analyses of the full range of issues covered in binding theory, such as ACD, paycheck sentences, etc.

References

Andrews, A. D. 2008a. Propositional glue and the correspondence architecture of LFG. URL: `http://arts.anu.edu.au/linguistics/People/AveryAndrews/Papers`; uploaded to Semantics Archive.

Andrews, A. D. 2008b. The role of PRED in LFG+glue. LFG-08 paper. URL: `http://arts.anu.edu.au/linguistics/AveryAndrews/papers`.

Asudeh, A. 2004. *Resumption as Resource Management*. PhD thesis, Stanford University, Stanford CA. `http://http-server.carleton.ca/~asudeh/` (viewed June 2008).

Babaev, A., and S. Solov'ev. 1982. A coherence theorem for canonical morphisms in cartesian closed categories. *Zap. Nauchn. Sem* LOM1 88:3–39, 236. also J. Soviet Math. 20:2263-2279.

Barwise, J., and R. Cooper. 1981. Generalized quantifiers and natural language. *Linguistics and Philosophy* 4:159–219.

Dalrymple, M. 2001. *Lexical Functional Grammar*. Academic Press.

Dalrymple, M., R. M. Kaplan, J. T. Maxwell, and A. Zaenen (Eds.). 1995. *Formal Issues in Lexical-Functional Grammar*. Stanford, CA: CSLI Publications. URL: `http://standish.stanford.edu/bin/detail?fileID=457314864`.

de Groote, P. 1999. An algebraic correctness criterion for intuitionistic multiplicative proof-nets. *TCS* 115–134. URL: `http://www.loria.fr/~degroote/bibliography.html` (viewed June 2008).

de Groote, P., and C. Retoré. 1996. On the semantic reading of proof-nets. In G. G.-J. Kruijff and D. Oehrle (Eds.), *Formal Grammar*, 57–70, FOLLI Prague, August. URL: `http://citeseer.ist.psu.edu/degroote96semantic.html`.

Dougherty, D. J. 1993. Closed categories and categorial grammar. *Notre Dame Journal of Formal Logic* 34(1):36–49.

Halvorsen, P.-K., and R. M. Kaplan. 1988. Projections and semantic description in Lexical-Functional Grammar. In *Proceedings of the International Conference on Fifth Generation*

Computer Systems, 1116–1122, Tokyo. Institute for New Generation Computer Technology. also published in Dalrymple et al. (1995), pp 279-292.

Jackendoff, R. S. 1973. The base rules for prepositional phrases. In S. R. Anderson and P. Kiparksy (Eds.), *A Festschrift for Morris Halle*, 345–357. Holt, Rinehard and Winston.

Jacobson, P. 1999. Towards a variable-free semantics. *Linguistics and Philosophy* 22:117–184.

Jacobson, P. 2002. The (dis)organization of the grammar: 25 years. *Linguistics and Philosophy* 25:601–626.

Jacobson, P. 2005. Direct compositionality and variable-free semantics: the case of 'principle b' effects. In *Direct Compositionality*. Oxford University Press.

Jaśkowski, S. 1963. Über Tautologien, in welchen keine Variable mehr als zweimal vorkommt. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 9:219–228.

Kaplan, R. M. 1995. The formal architecture of LFG. In M. Dalrymple, R. M. Kaplan, J. T. Maxwell, and A. Zaenen (Eds.), *Formal Issues in Lexical-Functional Grammar*, 7–27. CSLI Publications.

Kokkonidis, M. 2008. First order glue. *Journal of Logic, Language and Information* 17:43–68. First distributed 2006; URL: `http://citeseer.ist.psu.edu/kokkonidis06firstorder.html`.

Lamarche, F. 1994. Proof nets for intuitionistic linear logic 1: Essential nets. Technical Report, Imperial College.

Lambek, J., and P. J. Scott. 1986. *Introduction to Higher Order Categorical Logic*. Cambridge University Press.

Lev, I. 2007. *Packed Computation of Exact Meaning Representations using Glue Semantics (with automatic handling of structural ambiguities and advanced natural language constructions)*. PhD thesis, Stanford University. URL: `http://www.geocities.com/iddolev/pulc/current_work.html` (viewed June 2008).

Marantz, A. 1984. *On the Nature of Grammatical Relations*. Cambridge MA: MIT Press.

Melliès, P.-A. 2008. Categorical semantics of linear logic: a survey. unpublished draft, URL: `http://www.pps.jussieu.fr/~mellies/`.

Moot, R. 2002. *Proof-Nets for Linguistic Analysis*. PhD thesis, University of Utecht. URL: `http://www.labri.fr/perso/moot/` (viewed June 2008), also `http://igitur-archive.library.uu.nl/dissertations/1980438/full.pdf`.

Perrier, G. 1999. Labelled proof-nets for the syntax and semantics of natural languages. *L.G. of the IGPL* 7:629–655. URL: `http://www.loria.fr/~perrier/papers.html`.

Pollard, C. 2008. Hyperintensions. *Journal of Logic and Computation* 18:257–282. URL: `http://www.ling.ohio-state.edu/~pollard/cvg/hyper.pdf` (viewed June 2008).

Sag, I. 1976. *Deletion and Logical Form*. PhD thesis, MIT.

Tiede, H.-J. 1999. *Deductive Systems and Grammars: Proofs as Grammatical Structures.* PhD thesis, Indiana University. URL: `www.iwu.edu/~htiede/papers/pdf/diss.pdf` (seen Dec 2008).

Troelstra, A. S. 1992. *Lectures on Linear Logic.* Stanford CA: CSLI Publications.

Troelstra, A., and H. Schwichtenberg. 2000. *Basic Proof Theory.* Cambridge University Press. 2nd edition.

Wells, J., and B. Jakobowski. 2003. Graph-based proof counting and enumeration with applications for program fragment synthesis. In *Logic Based Program Synthesis and Transformation*, 262–277. Springer.